

# Символы и строки в языке Pascal

`char` `string`

Урок

# Символы в языке Pascal

# Тип Char

В памяти компьютера могут храниться **символы**.

Каждый символ кодируется кодом ASCII и занимает **один байт** памяти.

Для данных, представляющих собой одиночный символ, используется тип данных **char**.

Значение символьных данных — это символ, взятый в **апострофы**, например, **'А'** , **'?''** , **'5'**.

Знак апострофа при записи представляется **двумя** апострофами.

# Char – порядковый тип

Символьный тип относится к порядковым типам данных.

Из этого следует:

- СИМВОЛЫ — **упорядоченное множество**;
- у каждого символа в этом множестве есть свой

**порядковый номер**;

- между символами **работает соотношение**

«следующий—предыдущий».

# Код символа

Порядковый номер символа — это его код, десятичное значение которого лежит в диапазоне **от 0 до 255**.

В языке Pascal для представления символов используется код **ASCII**. Например, десятичный код латинской буквы **'A'** равен **65**, а цифры **'5'** — **53**.

# Функция Ord

**Ord (x)** — функция от аргумента порядкового типа, которая возвращает порядковый номер значения **x** в этом типе данных.

Если **x** — символьная величина, то результатом функции будет десятичный код **x** в кодовой таблице.

Например:

**Ord ( ' A ' ) = 65**

**Ord ( ' 5 ' ) = 53**

# Функция Chr

**Chr (x)** — функция от целочисленного аргумента, результатом которой является символ с кодом, равным **x**.

Например:

**Chr (65) = ' A '**

**Chr (53) = ' 5 '**

# Принцип последовательного кодирования алфавитов

В любой кодовой таблице выполняется принцип последовательного кодирования букв латинского (английского) алфавита, десятичных цифр и букв русского алфавита (кроме буквы **Ё**).

Это обстоятельство часто используют в программах обработки символьной информации.



# Сравнение символов

Чем больше значение кода, тем символ считается больше. Истинными являются следующие отношения:

'A' < 'B' ,

'z' > 'Y' ,

'a' > 'A' .

Если истинно логическое выражение

$(C \geq 'A') \text{ and } (C \leq 'Z')$  значением символьной

переменной **C** является заглавная латинская буква.

Если истинно логическое выражение

$(C \geq '0') \text{ and } (C \leq '9')$  значением символьной

переменной **C** является цифра.

# Массив символов

Символы могут объединяться в массивы. Каждому элементу массива, как и числовым данным, соответствует порядковый номер, а имя элемента состоит из имени всего массива и его номера.

Это массив символов:

```
var B:array[1..50] of char;
```

Посмотреть пример 2 (стр.183)

# Строки в языке Pascal

# Чем плох массив символов?

Это массив СИМВОЛОВ:

```
const N=20;  
var B: array[1..N] of char;
```

**Что имеем:**

- каждый СИМВОЛ – отдельный объект;

**Что нужно:**

- обрабатывать последовательность СИМВОЛОВ как единое целое
- последовательность СИМВОЛОВ должна иметь длину

# Символьные строки

```
var s:string;
```

длина строки

1

s[3]

s[4]

↓  
255



← рабочая часть →

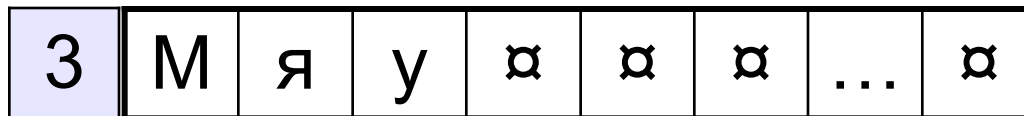
s[1]

s[2]

```
var s:string[20];
```

1

20



```
var n: integer;
```

Длина строки:

```
n:=length(s);
```

# Задача

**Задача:** ввести строку с клавиатуры и заменить все буквы «а» на буквы «б».

```
program qq;  
var s:string;  
    i:integer;  
begin  
    writeln('Введите строку');  
    readln(s);  
    for i:=1 to Length(s) do  
        if s[i]='a' then s[i]:='б';  
    writeln(s);  
end.
```

ВВОД строки

длина строки

ВЫВОД строки

# Операции со строками

```
var s, s1, s2: string;
```

**Присваивание нового значения:**

```
s := 'Вася';
```

**Объединение:** добавить одну строку в конец другой.

```
s1 := 'Привет';  
s2 := 'Вася';  
s := s1 + ', ' + s2 + '!';
```

'Привет, Вася!'

**Подстрока:** скопировать часть строки в другую строку.

```
s := '123456789';
```

с 3-его символа

6 штук

```
s1 := Copy(s, 3, 6);
```

```
s2 := Copy(s1, 2, 3);
```

'345678'

'456'

# Удаление и вставка

## Удаление части строки:

6 штук

```
s := '123456789' ;  
Delete (s, 3, 6) ;
```

'12~~345678~~9'  
'129'

строка  
меняется!

с 3-его символа

## Вставка в строку:

начиная с 3-его символа

```
s := '123456789' ;  
Insert ('ABC', s, 3) ;
```

'12ABC3456789'

что  
вставляем

куда  
вставляем

```
Insert ('Q', s, 5) ;
```

'12ABQC3456789'



# Поиск в строке

Поиск в строке:

s[3]

```
var n: integer;
```

```
s := 'Здесь был Вася.' ;  
n := Pos('е', s) ;  
if n > 0 then  
    writeln('Буква е - это s[' , n , ']')  
else writeln('Не найдено') ;  
n := Pos('Вася', s) ;  
s1 := Copy(s, n, 4) ;
```

3

n = 11

## Особенности:

- функция возвращает номер символа, с которого начинается образец в строке
- если слова нет, возвращается 0
- поиск с начала (находится **первое** слово)

# Примеры

```
s:='Вася Петя Митя';  
n:=Pos('Петя',s);  
Delete(s,n,4);  
Insert('Лена',s,n);
```

6

'Вася Митя'

'Вася Лена Митя'

```
s:='Вася Петя Митя';  
n:=length(s);  
s1:=Copy(s,1,4);  
s2:=Copy(s,11,4);  
s3:=Copy(s,6,4);  
s:=s3+s1+s2;  
n:=length(s);
```

14

'Вася'

'Митя'

'Петя'

'ПетяВасяМитя'

12

# Преобразования (классические способы языка)

## Из строки в число:

```
s:='123';  
Val(s,N,r); {N=123}  
  {r=0, если ошибки не было  
   r - номер ошибочного символа}  
s:='123.456';  
Val(s,X,r); {X=123.456}
```

```
var N,r: integer;  
    X: real;  
    s: string;
```

## Из числа в строку:

```
N:=123;  
Str(N,s);      {'123'}  
X:=123.456;  
Str(X,s);      {'1.234560E+002'}  
Str(X:10:3,s); {'  123.456'}
```

# Задача

**Задача:** Ввести имя, отчество и фамилию. Преобразовать их к формату «фамилия-инициалы».

**Пример:**

**Имя, отчество и фамилия:**

**Василий Алибабаевич Хрюндиков**

**Результат:**

**Хрюндиков В. А.**

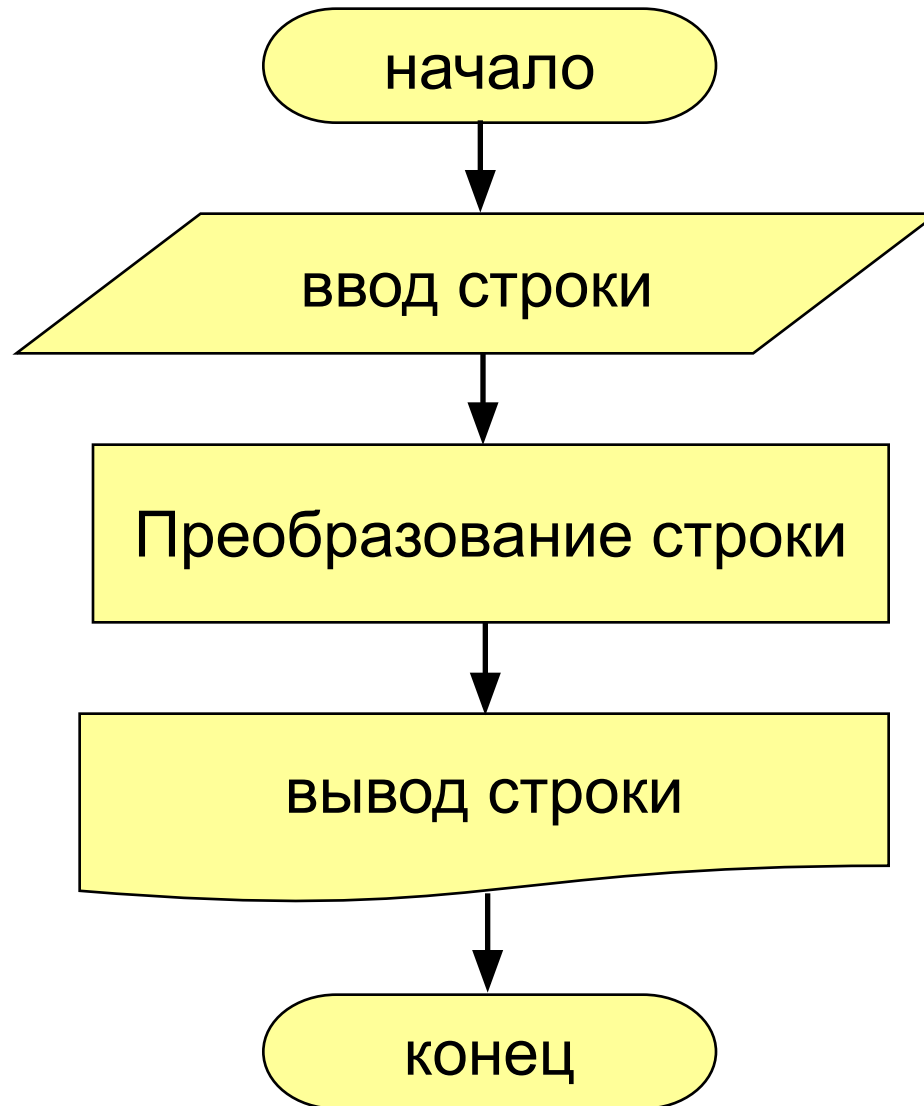
**Алгоритм:**

- найти первый пробел и выделить имя
- удалить имя с пробелом из основной строки
- найти первый пробел и выделить отчество
- удалить отчество с пробелом из основной строки
- «склеить» фамилию, первые буквы имени и фамилии, точки, пробелы...

# Задание

1. Создать в **личной папке** файл **strings.pas**.  
Составить программу, которая преобразует строку в формате **<Имя Отчество Фамилия>** в строку в формате **<Фамилия И. О.>**
2. Выполнить и **отладить** программу в среде **Turbo**.  
Закрыть среду **Turbo с сохранением** файла.
3. Протестировать **скомпилированную** программу (**STRINGS.EXE** в личной папке).
4. Продемонстрировать работу программы **при учителе**.

# Укрупнённый алгоритм



# Шпаргалка

```
Program .....  
var s,name,otch:string;  
    n:integer;  
begin  
    //Ввод строки  
    .....  
    //Вычисление новой строки  
    n:=Pos(' ',s);           // найти первый пробел  
    name:=Copy(s,1,n-1);     // скопировать имя  
    Delete(s,1,n);           // удалить имя и пробел  
    n:=Pos(' ',s);           // найти первый пробел  
    otch:=Copy(s,1,n-1);     // скопировать отчество  
    Delete(s,1,n);           // осталась фамилия  
    s:=s+' '+name[1]+'.'+otch[1]+'.';  
    //Вывод строки  
    .....  
end.
```

# Работаем за компьютером

